Remote Controlled Robot

David Pajar

Rimon Sawa

**Goal:**

* Control a robot using RF control
* Implement an AVR and RF module
* Move robot forward, reverse, left, right and stop

**Project:**

Remote controlled robot based on ATmega328P + RF + PC

The goal of this project will be to control a zumo robot (provided fully assembled) using RF control. The zomo robot will be equipped and controlled with an AVR + RF module. The host system will consist of an AVR+RF module connected to the PC through UART interface. Commands will be issued from the PC through the terminal or visual GUI to move the robot move forward, reverse, left, right and stop.

# I. COMPONENTS

## A. ATmega328p

The ATmega328p Xplained Mini is an 8-bit microcontroller from Atmel that can be programmed with either Atmel Studio or Arduino IDE using C/C++, AVR assembly, or Arduino. The code will allow users to manipulate the board’s 32 GPIOs, 6 PWMs, 3 timer/counters, 1kB EEPROM, 2kB SDRAM, I2C interface, SPI interface, 10-bit ADCs, and up to an internal 16 MHz clock.

## B. RF Remote

This 4-button keyfob remote goes with our three basic 315MHz RF receiver modules. It will work with all of them, sending out one of four commands that match up with the four outputs. Its small and light weight and will work up to 25 feet away depending on line-of-sight and obstructions. Since this is just a transmitter, if you have multiple receivers, it will turn all of them on and off at the same time (there is no sub-addressing)

## C. Gear motors

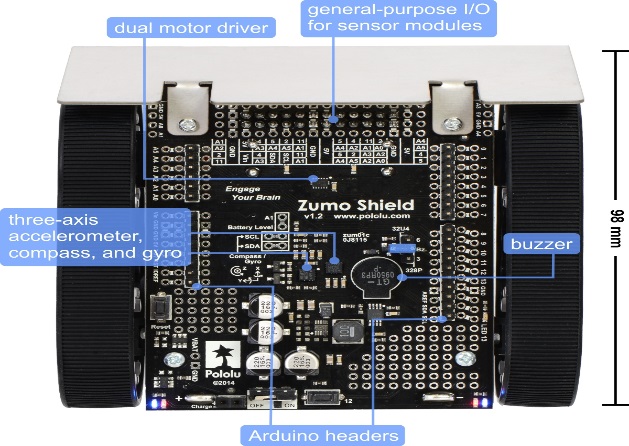
## D. 2 \* 3.7V battery

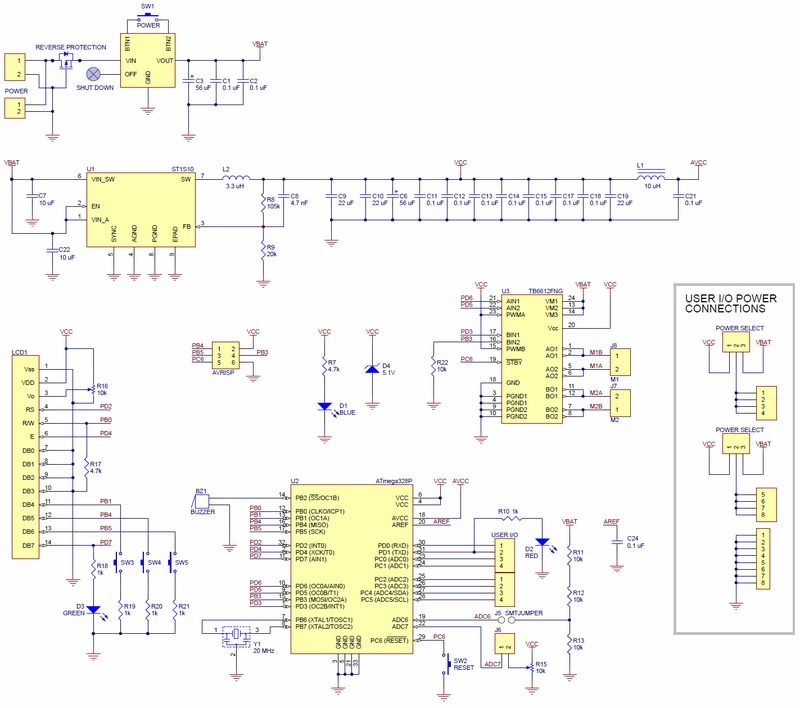
To power microcontroller as well as motor module.

**E. Model X Motor Driver module**

Module is used to connect all four DC motors and to be able to control them.

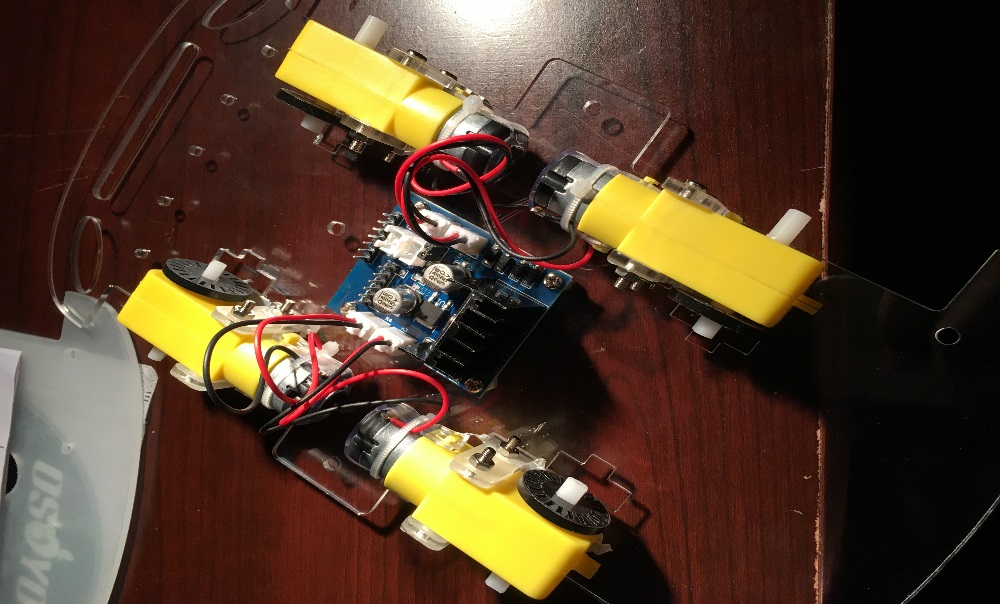
III. Schematics





# IV. IMPLEMENTATION

Implementing the RF module to work with the atmega328p board. Once configured, we are able to control the robot.

V. SNAPSHOTS AND SCREENSHOTS 

VI. Code

Main:

#include <IRremote.h>

#include "configuration.h"

 IRrecv IR(IR\_PIN);  //   IRrecv object  IR get code from IR remoter

 decode\_results IRresults;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*motor control\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void go\_Advance(void)  //Forward

{

  digitalWrite(dir1PinL, HIGH);

  digitalWrite(dir2PinL,LOW);

  digitalWrite(dir1PinR,HIGH);

  digitalWrite(dir2PinR,LOW);

  analogWrite(speedPinL,255);

  analogWrite(speedPinR,255);

}

void go\_Left(int t=0)  //Turn left

{

  digitalWrite(dir1PinL, HIGH);

  digitalWrite(dir2PinL,LOW);

  digitalWrite(dir1PinR,LOW);

  digitalWrite(dir2PinR,HIGH);

  analogWrite(speedPinL,200);

  analogWrite(speedPinR,200);

  delay(t);

}

void go\_Right(int t=0)  //Turn right

{

  digitalWrite(dir1PinL, LOW);

  digitalWrite(dir2PinL,HIGH);

  digitalWrite(dir1PinR,HIGH);

  digitalWrite(dir2PinR,LOW);

  analogWrite(speedPinL,200);

  analogWrite(speedPinR,200);

  delay(t);

}

void go\_Back(int t=0)  //Reverse

{

  digitalWrite(dir1PinL, LOW);

  digitalWrite(dir2PinL,HIGH);

  digitalWrite(dir1PinR,LOW);

  digitalWrite(dir2PinR,HIGH);

  analogWrite(speedPinL,255);

  analogWrite(speedPinR,255);

  delay(t);

}

void stop\_Stop()    //Stop

{

  digitalWrite(dir1PinL, LOW);

  digitalWrite(dir2PinL,LOW);

  digitalWrite(dir1PinR,LOW);

  digitalWrite(dir2PinR,LOW);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*detect IR code\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void do\_IR\_Tick()

{

  if(IR.decode(&IRresults))

  {

    if(IRresults.value==IR\_ADVANCE)

    {

      Drive\_Num=GO\_ADVANCE;

    }

    else if(IRresults.value==IR\_RIGHT)

    {

       Drive\_Num=GO\_RIGHT;

    }

    else if(IRresults.value==IR\_LEFT)

    {

       Drive\_Num=GO\_LEFT;

    }

    else if(IRresults.value==IR\_BACK)

    {

        Drive\_Num=GO\_BACK;

    }

    else if(IRresults.value==IR\_STOP)

    {

        Drive\_Num=STOP\_STOP;

    }

    IRresults.value = 0;

    IR.resume();

  }

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*car control\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void do\_Drive\_Tick()

{

    switch (Drive\_Num)

    {

      case GO\_ADVANCE:go\_Advance();JogFlag = true;JogTimeCnt = 1;JogTime=millis();break;//if GO\_ADVANCE code is detected, then go advance

      case GO\_LEFT: go\_Left();JogFlag = true;JogTimeCnt = 1;JogTime=millis();break;//if GO\_LEFT code is detected, then turn left

      case GO\_RIGHT:  go\_Right();JogFlag = true;JogTimeCnt = 1;JogTime=millis();break;//if GO\_RIGHT code is detected, then turn right

      case GO\_BACK: go\_Back();JogFlag = true;JogTimeCnt = 1;JogTime=millis();break;//if GO\_BACK code is detected, then backward

      case STOP\_STOP: stop\_Stop();JogTime = 0;break;//stop

      default:break;

    }

    Drive\_Num=DEF;

   //keep current moving mode for  200 millis seconds

    if(millis()-JogTime>=200)

    {

      JogTime=millis();

      if(JogFlag == true)

      {

        stopFlag = false;

        if(JogTimeCnt <= 0)

        {

          JogFlag = false; stopFlag = true;

        }

        JogTimeCnt--;

      }

      if(stopFlag == true)

      {

        JogTimeCnt=0;

        stop\_Stop();

      }

    }

}

void setup()

{

  pinMode(dir1PinL, OUTPUT);

  pinMode(dir2PinL, OUTPUT);

  pinMode(speedPinL, OUTPUT);

  pinMode(dir1PinR, OUTPUT);

  pinMode(dir2PinR, OUTPUT);

  pinMode(speedPinR, OUTPUT);

  stop\_Stop();

  pinMode(IR\_PIN, INPUT);

  digitalWrite(IR\_PIN, HIGH);

  IR.enableIRIn();

}

void loop()

{

  do\_IR\_Tick();

  do\_Drive\_Tick();

}

Implementation File:

#define dir1PinL 2 //Motor direction

#define dir2PinL 4 //Motor direction

#define speedPinL 6 // Needs to be a PWM pin to be able to control motor speed

#define dir1PinR 7 //Motor direction

#define dir2PinR 8 //Motor direction

#define speedPinR 5 // Needs to be a PWM pin to be able to control motor speed

#define IR\_PIN 3 //IR receiver Signal pin connect to Arduino pin 3

#define IR\_ADVANCE 0x00FF18E7 //code from IR controller "▲" button

#define IR\_BACK 0x00FF4AB5 //code from IR controller "▼" button

#define IR\_RIGHT 0x00FF5AA5 //code from IR controller ">" button

#define IR\_LEFT 0x00FF10EF //code from IR controller "<" button

#define IR\_STOP 0x00FF38C7 //code from IR controller "OK" button

#define IR\_turnsmallleft 0x00FFB04F //code from IR controller "#" button

enum DN

{

GO\_ADVANCE, //go forward

GO\_LEFT, //left turn

GO\_RIGHT,//right turn

GO\_BACK,//backward

STOP\_STOP,

DEF

}Drive\_Num=DEF;

bool stopFlag = true;//set stop flag

bool JogFlag = false;

uint16\_t JogTimeCnt = 0;

uint32\_t JogTime=0;

uint8\_t motor\_update\_flag = 0;

# VII. LINKS

David Pajar github: [www.github.com/pajard1](http://www.github.com/pajard1)

Rimon Sawa github: [www.github.com/rsawa91](http://www.github.com/rsawa91)

Youtube Video: <https://youtu.be/QFLYzRwwKnE>

## VIII. CONCLUSION

We were able to control the robot using the atmega328p along with the RF module. We displayed the movements moving forward, backward, left, right and stopping.